LONDON METROPOLITAN UNIVERSITY

ITAHARI INTERNATIONAL COLLEGE

Module Code & Module Title

CS4001NT Programming

Assessment Weightage & Type

30% Individual Coursework

Year and Semester

2021-22 Autumn 2

Student Name: Amshul Dhungel

Group: L1C3 A2

London Met ID: 2104982

College ID: NP05CP4A210044

Assignment Due Date: 10th October 2022, 4:00 PM

Assignment Submission Date: 26th September 2022, 11:00pm

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a marks of zero will be awarded.

Contents

List of Figures

List of Table

1. INTRODUCTION

One of the best instrument rental and sales companies in my country is SarangiSansar, serving millions of Nepalese customers with instruments like the guitar, madal, and others. In 2022, it was discovered. Yes, you read that correctly—we rent out instruments for a variety of uses. When necessary for important Nepalese occasions like bartamanda, parties, and wedding ceremonies, they are marketed to the general public and hired out.

Java is a programming language that compile and solve the problem. In this module we have use Java swing for creating a simple GUI frame in which we record information of the selling and renting instrument. A component of the Java Foundation Classes (JFC) used to develop window-based applications is the Java Swing tutorial. (edureka, 2022). It is totally developed in Java and constructed on top of the AWT (Abstract Windowing Toolkit) API.

Classes supporting the Java Swing API are available in the javax.swing package, including JButton, JTextField, JTextArea, JRadioButton, JCheckbox, JMenu, JColorChooser, etc.

Swing in Java:

A lightweight GUI toolkit for constructing window-based applications called Swing in Java contains a large selection of widgets. It's a component of the JFC (Java Foundation Classes). It is totally java-based and built on top of the AWT API. In contrast to AWT, it features lightweight components and is platform independent.

Since there are already GUI components like buttons, checkboxes, etc., creating applications is made simpler. We do not have to start from scratch, which is advantageous.

Figure 1: Swing in Java (edureka, 2022)

GUI (Graphical User Interface)

Java's GUI (Graphical User Interface) is a simple tool for creating visual experiences for Java programs. It is mostly formed of graphical elements that allow the user to interact with a program, such as buttons, labels, windows, etc. A GUI is crucial in creating user-friendly interfaces for Java programs.

Software used for the project:

For this coursework I have used different types of software like BlueJ, Snipping tool, draw.io etc. so some of them are described below:

Figure 2 BlueJ

BlueJ is a Java integrated development environment (IDE) that was created primarily for instructional purposes but is also appropriate for development of software on a modest scale. It works Using JDK (Java Development Kit). BlueJ was created to assist with the the study and instruction of object-oriented Programming and its structure are distinct from additional settings for development, including result. (BlueJ, 2022)

Snipping Tools:

Snipping Tool is a tool that allows users to capture screenshots of either their entire screen or a specific area of it as needed. Additionally, it enables users to save screenshots as images.

A snipping tool can be used by simply selecting Start and typing "Snipping Tool," or by pressing "Windows+Shift+S."

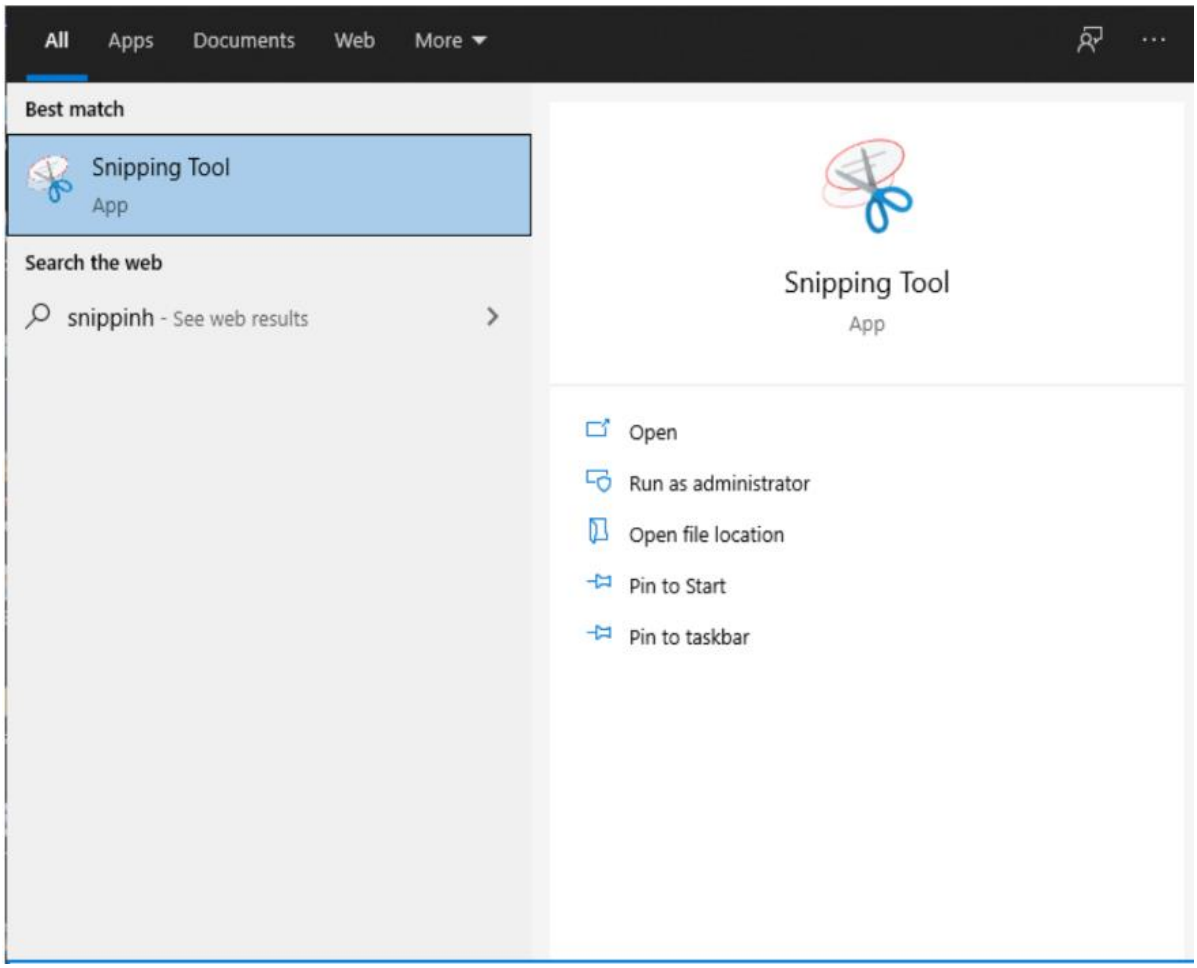


Figure 3 Snipping tools

2. About Project

Field Summary

Fields	
Modifier and Type	**Field**
(package private) static int	**EMPTY**
(package private) java.util.ArrayList<Instrument>	**instrumentList**
(package private) int	**integer**
(package private) static int	**INVALID**
(package private) int	**num**
(package private) java.lang.String	**str**

Table 1: Field Summary

Constructor Summary

Constructors
Constructor
SarangiSansar()

a. Class-Diagram

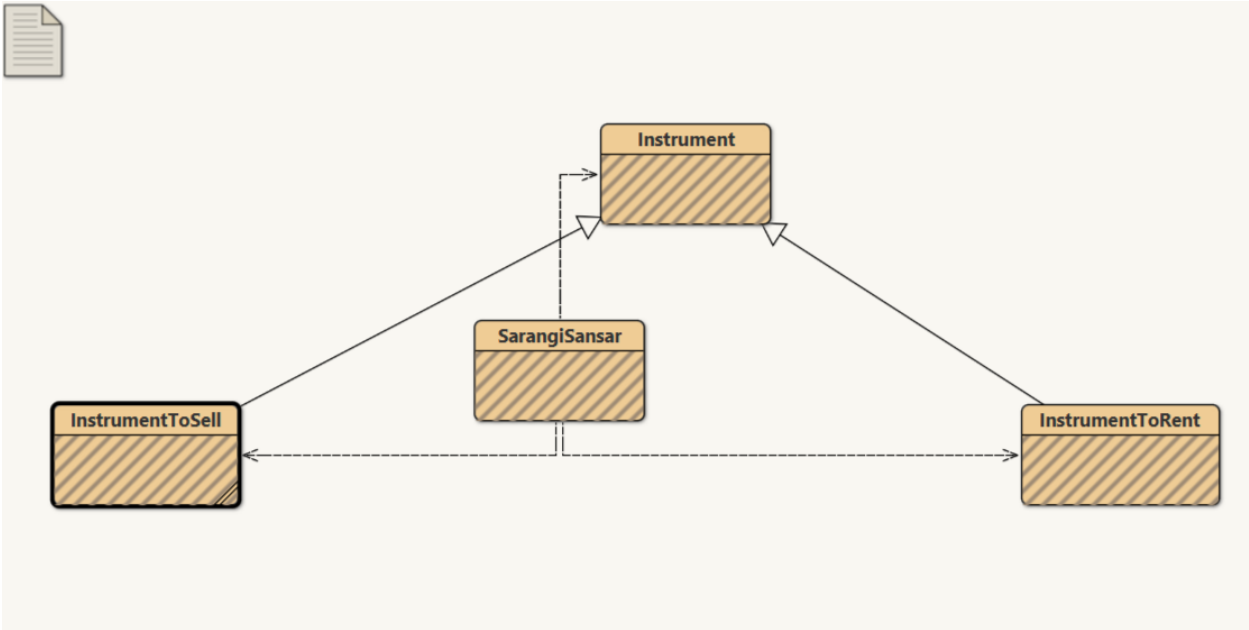


Figure 4: Class Diagram

SarangiSansar	
Private	Public
instrumentList rentsellPanel: rentSellPanel rentsellpanel1: subpanel6 homepanel JLabel JTextField JButton JComboBox	addInstrumentForSell() addinstrumenttorentBtn() clearRent() clearSell() getChargePerDay() getDiscountPercent() getInstrumentRentName() getPanNoRent() getPanNoSell() getPrice() getrentedNoOfDays() rentInstrument() returnInstrument() sellInstrument()

c. A short description of what each method does

Modifier and Type	Method	Description
void	addInstrumentForSell()	This method handles the JButton part of Instrument to add the instrument in Go to sell frame
void	addinstrumenttorentBtn ()	This method handles the JButton part of Instrument to add the instrument in Go to Rent frame
void	clearRent()	This method is used to handle clear buttonwhich helps to clear all the text written in Textfield in Go to Rent Frame
void	clearSell()	This method is used to handle clear button which helps to clear all the text written in TextField in Go to Rent Frame
int	getChargePerDay()	This is return type method which is used for getting charge per day of the instrument customer rented.
int	getDiscountPercent()	This is return type method which is used for getting discount percent of the instrument customer made while buying the instrument.
java.lang.String	getInstrumentRentName()	It is used for extracting the value of'InstrumentName' in rent panel of the rent and return the value of 'InstrumentName'.
int	getPanNoRent()	This is return type method which is used for getting pan number of instrument to rent.
int	getPanNoSell()	This method is used for getting pan number of sell instrument from Go to sell frame

int	getPrice()	This method is return type which is used for getting price from selling the instrument in Go to sell frame
int	getrentedNoOfDays()	This method is return type which is used for getting number of days customer takes when rented instrument
static void	main (java.lang.String[] args)	This is main method where we display the output part of the coursework
void	rentInstrument()	This is a void return type which helps to set the value of customer name, phone number, pan number, renting date, returning date and number of days.
void	returnInstrument()	This is a void return type which helps to know the return of instrument after knowing whether rented of instrument is proceed to the costumer
void	sellInstrument()	This is a void return type which helps to set the value of customer name, customer phone number, pan number, selling date and discount percent

Table 2: Short description of method

b. Pseudocode

A straightforward method of writing programming code in English is pseudocode. There is no real programming language used in pseudo code. Before you really write the computer code in a certain language, it employs brief words to do so. after you are aware of the purpose of the program then you can use pseudocode to generate statements to achieve what you want the outcomes that your program requires.

It is simpler to create programs using pseudo code. Programs can be lengthy and complex; the key is preparation. Before creating a single line of code in a language, programs were for years mapped out using flowcharts. However, they were complicated to change, and as programming languages advanced, it became more difficult to display every aspect of a program on a flowchart, making it difficult to see errors without knowing how a program functions in its entirety. That is when the need for pseudo code increases.

Below is the pseudocode of the following program:

```
Import *

Create Class SarangiSansar

    List InstrumentList

    Var Empty = -1

    Var invalid = -2

    Integer num = 1

    String str = " "
```

Integer integer = 1

Create Checkbox addarkmode

Create Frame rent, sell, homepage

Create Label

Create Textfield

Create Button

Create combo box

Method SarangiSansar

String days = Assign values

String months

String Years

Assign Color

Assign Font

Assign boundaries

Assign Location, size

Assign Background

Method actionperfomed

if integer % 2 Equals 0

Set background, set Foreground, setBackground

else

set background,

set foreground

Assign Frames, background, listener

Method EventListener(Mouse)

Set Background

Assign title, Frame, Text, Button, Label

Return title, Frame, Text, Button, Label

Method ActionListener

addButton

Method Main

Initiate SarangiSansar

Method void

Declare String InstrumentName, instrumentToRentText,customerNameRentText

phoneText, dateofrent, dateofreturn

int chargePerDay, panText, rentedNoOfDaysText

Declare isUnique Equal true

if num Equals 2

Declare num Equal 1

if str Equals invalid

Declare str Equal empty

```
        if instrumentName is Empty & chargePerDay Equal Empty
            JOptionPane.showMessageDialog(Warning Message)

        loop using foreach integer(int) Equals 0, size(instrumentList), addinteger
            if instrumentName.to Lowercase Equals(instrumentName.toLowerCase)
              Declare isUnique Equals false
            JOptionPane.showMessageDialog(Error Message)

        if isUnique & chargePerDay not Equal INVALID
            add InstrumentToRent
            JOptionPane.showMessageDialog(Information Message)

    Method get
        return rentname
        Declare ChargePerDay Equal INVALID

        if chargePerDay is Empty
            chargePerDay Equals Empty

        try
            if chargePerDay Equal or less 0
                JOptionPane.showMessageDialog(Warning Message)
```

chargePerDay Equal INVALID

number Equal 2

catch NumberFormatException number

JOptionPane.showMessageDialog(Error Message)

str Equal invalid

return

Method void(rentInstrument)

if instrumentToRentText is Empty or panText Equal EMPTY or rentedNoOfDaysText Equal EMPTY

or customerNameRentText is Empty

or phoneText is Empty

JOptionPane.showMessageDialog(Warning Message)

for negative

if num Equal 2

num Equal 1

if str Equal invalid

str Equal Empty

if size(instrumentList) less than 0

using for loop

integer Equal 0, size, addIntegers

if instrumentList.toLowercase equal instrumentToRentText.toLowercase

if instrumentList instance of InstrumentToRent

if isRented

isUnique Equal false

JOptionPane.showMessageDialog(Warning Message)

isUnique Equal false

JOptionPane.showMessageDialog(Information Message)

if moreThanZero Equal false

JOptionPane.showMessageDialog(Error Message)

Method void(clear rent)

setText

Method void(sellInstrument)

Declare String InstrumentName, instrumentToSellText,customerNameSellText

SellphoneText, dateofsell

int price, customerpanText, discountPercent

Declare isUnique Equal true

if num Equals 2

Declare num Equal 1

if str Equals invalid

Declare str Equal empty

if instrumentName is Empty & price Equal Empty

JOptionPane.showMessageDialog(Warning Message)

loop using foreach (integer(int) Equals 0, size(instrumentList), addinteger)

if instrumentName.to Lowercase Equals(instrumentName.toLowerCase)

Declare isUnique Equals false

JOptionPane.showMessageDialog(Error Message)

if isUnique & price not Equal INVALID

add InstrumentToRent

JOptionPane.showMessageDialog(Information Message)

Method get

return rentname

Declare PanNoSell Equal INVALID

if PanNoSell is Empty

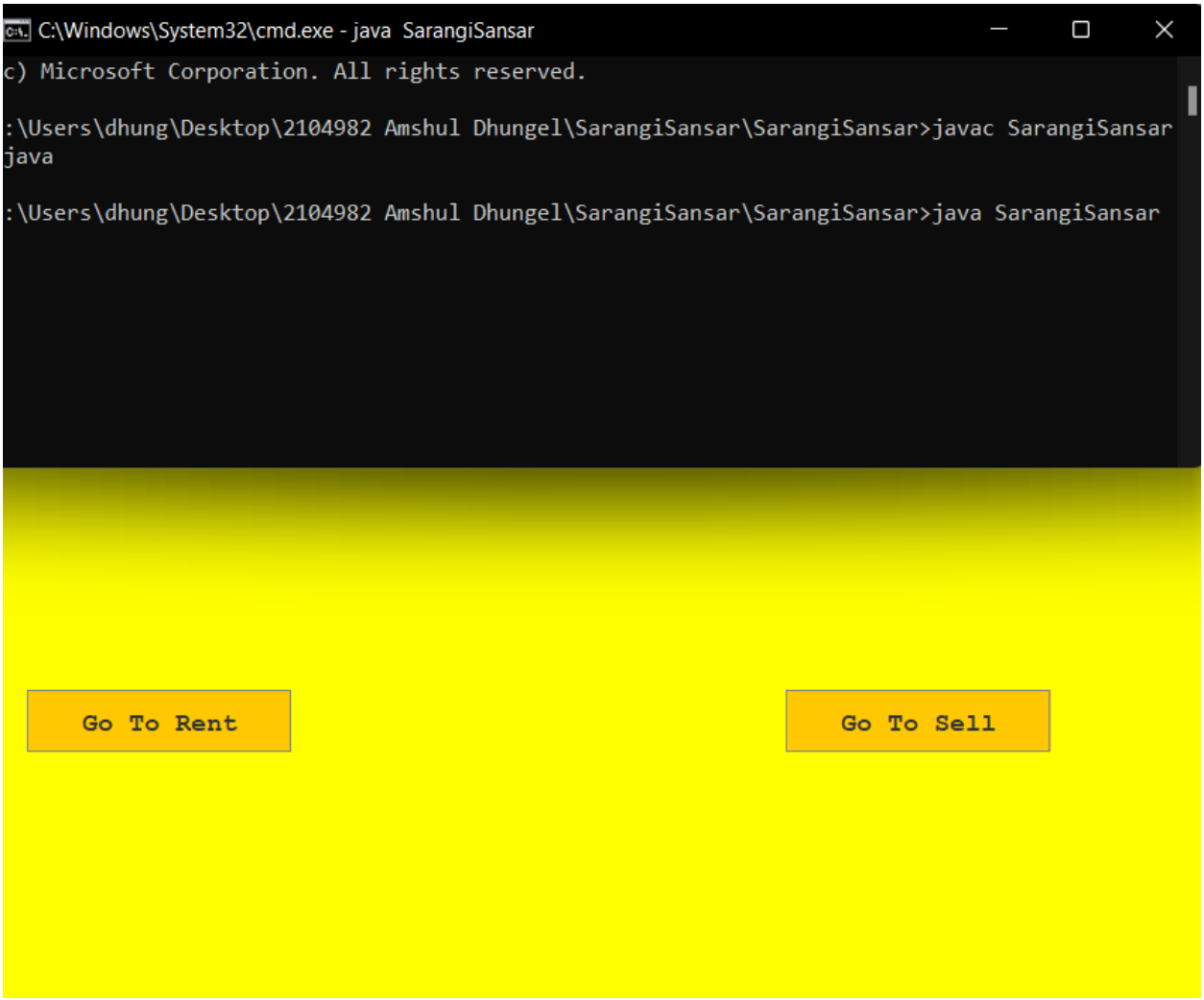
```
        PanNoSell Equals Empty


try
        if PanNoSell Equal or less 0
                JOptionPane.showMessageDialog(Warning Message)
                PanNoSell Equal INVALID
                number Equal 2
catch NumberFormatException number
        JOptionPane.showMessageDialog(Error Message)
        str Equal invalid
return
```

3. Testing

Test 1

Objective	To run code in command prompt
Action	On command prompt locate the folder where your file is stored. Then after Write: 1) javac SarangiSansar.java 2) java SarangiSansar
Expected Result	Homepanel or Homepage should be displayed
Actual Result	Home panel is displayed
Reference	Test is successful


C:\Windows\System32\cmd.exe - java SarangiSansar
c) Microsoft Corporation. All rights reserved.
:\Users\dhung\Desktop\2104982 Amshul Dhungel\SarangiSansar\SarangiSansar>javac SarangiSansar
java
:\Users\dhung\Desktop\2104982 Amshul Dhungel\SarangiSansar\SarangiSansar>java SarangiSansar
Go To Rent
Go To Sell

Test 2

a) Add Instrument for rent

Test	Adding instrument for rent
Objective	To add instrument for rent
Action	Fill the form in add panel in rent panel. Click the add button
Expected result	Message box: Instrument is added to rent successfully
Actual result	Message box: Instrument is added to rent successfully
Reference	Test successful

Table 3: Adding instrument for rent

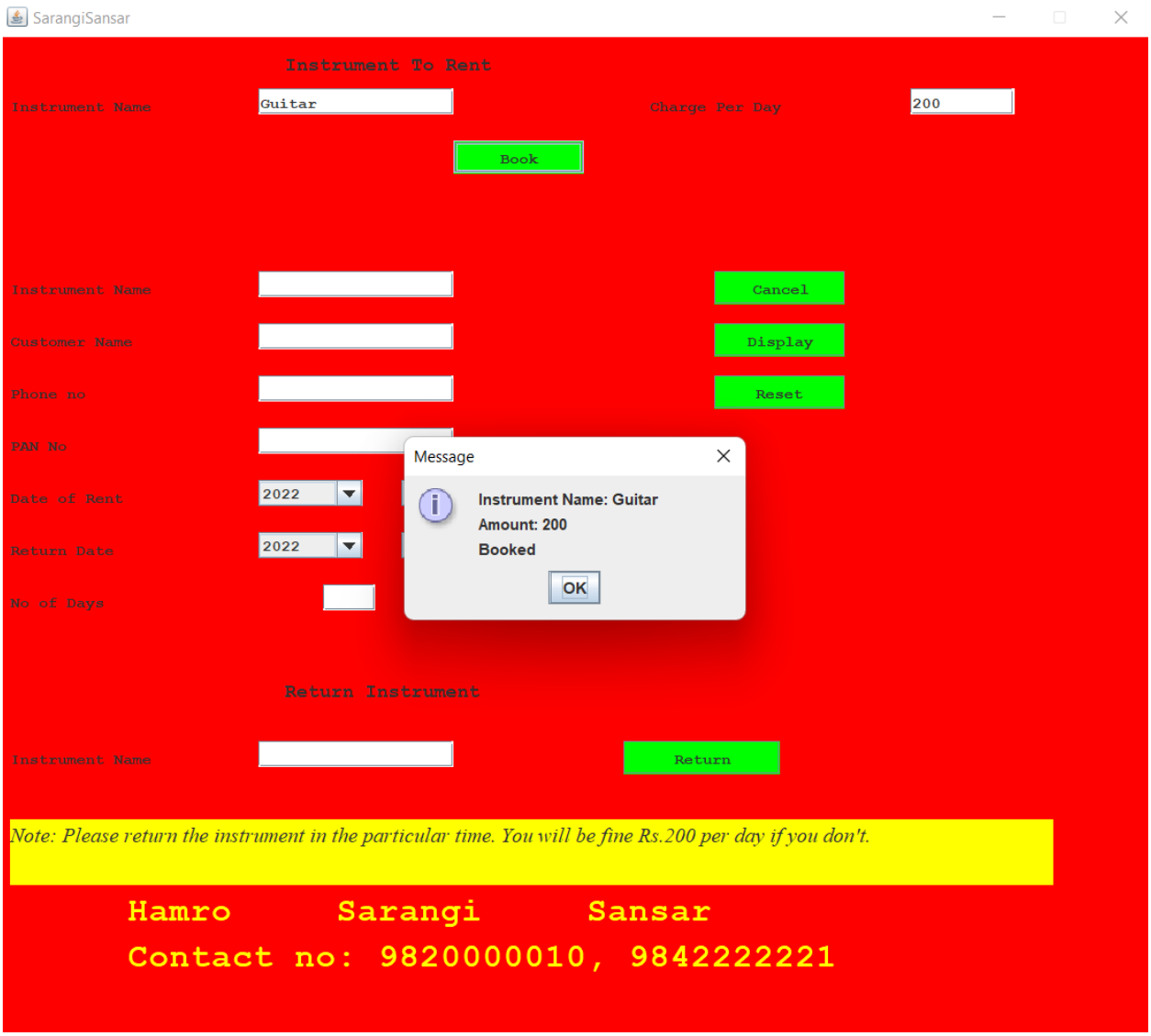


Figure 5: Add instrument for rent

b) Add Instrument for sell

Test	Adding instrument for sell
Objective	To add instrument for sell
Action	Fill the form in add panel in sell panel. Click the add button
Expected result	Message box: Successfully added to selling stock
Actual result	Message box: Successfully added to selling stock
Reference	Successful

Table 4: Add instrument for sell

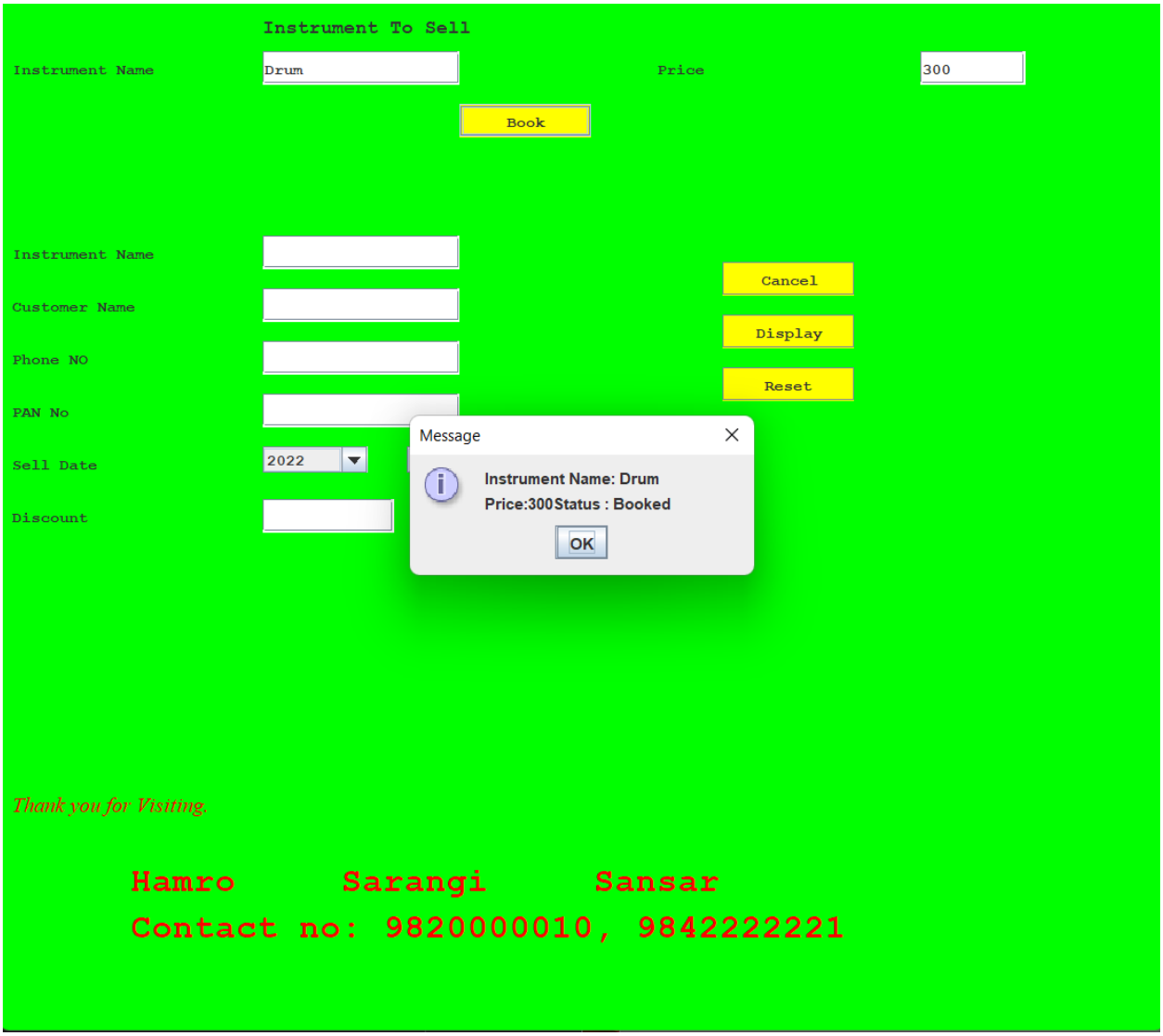


Figure 6 Add instrument for sell

c) Rent the Instrument

Test	Rent the Instrument
Objective	To rent the instrument
Action	Fill the form in add panel in rent panel(second box). Click the rent button
Expected result	Message box: Successfully added to renting stock
Actual result	Message box: Successfully added to renting stock
Reference	Successful

Table 5: Adding instrument for rent

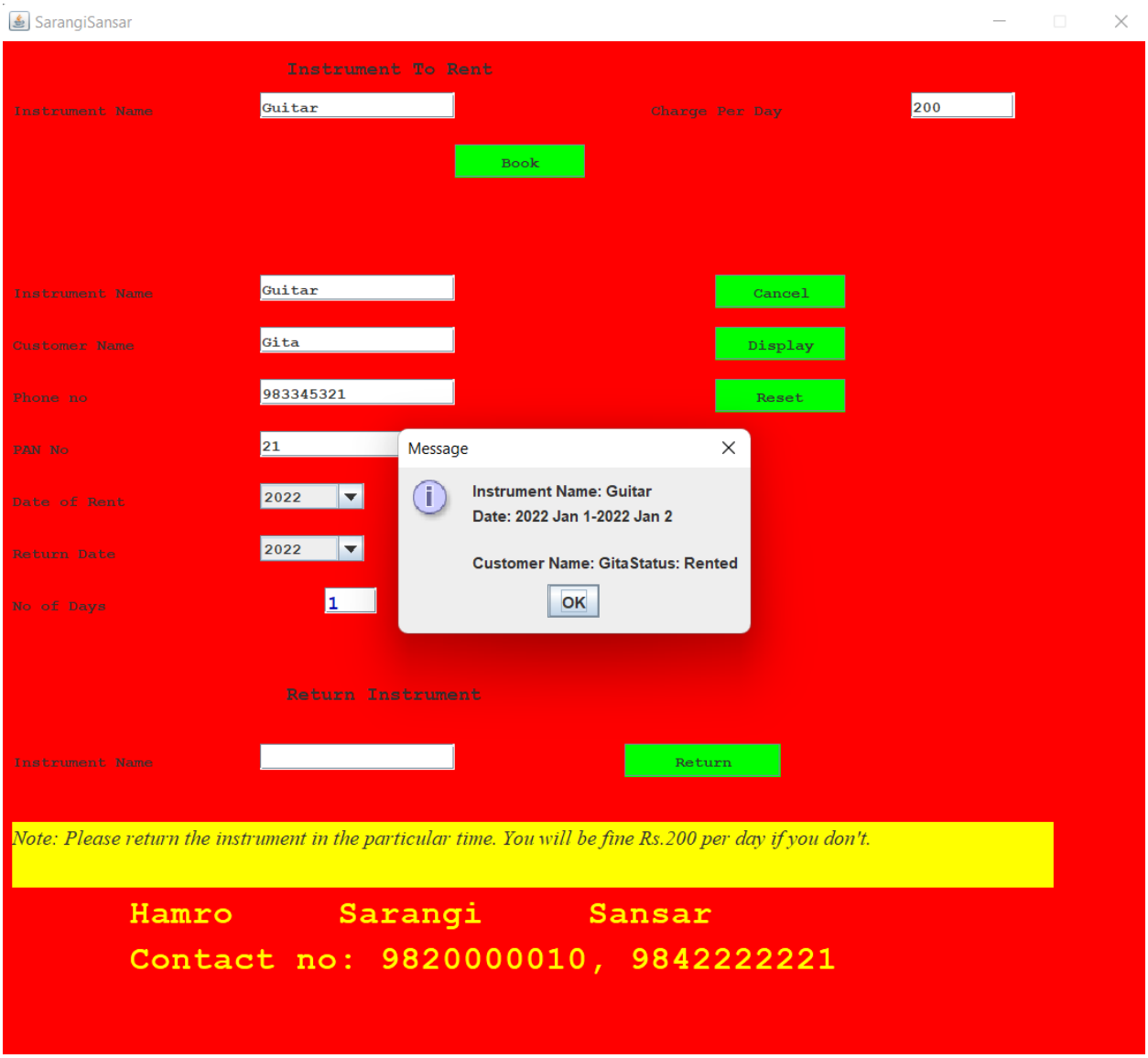


Figure 7: Adding instrument for rent

d) Sell the Instrument

Test	Sell the Instrument
Objective	To sell the instrument
Action	Go to sell. Fill the form in add panel in sell panel(second box). Click the sell button
Expected result	Instrument should be sold
Actual result	Message box: Successfully sold
Reference	successful

Table 6: Sell the instrument

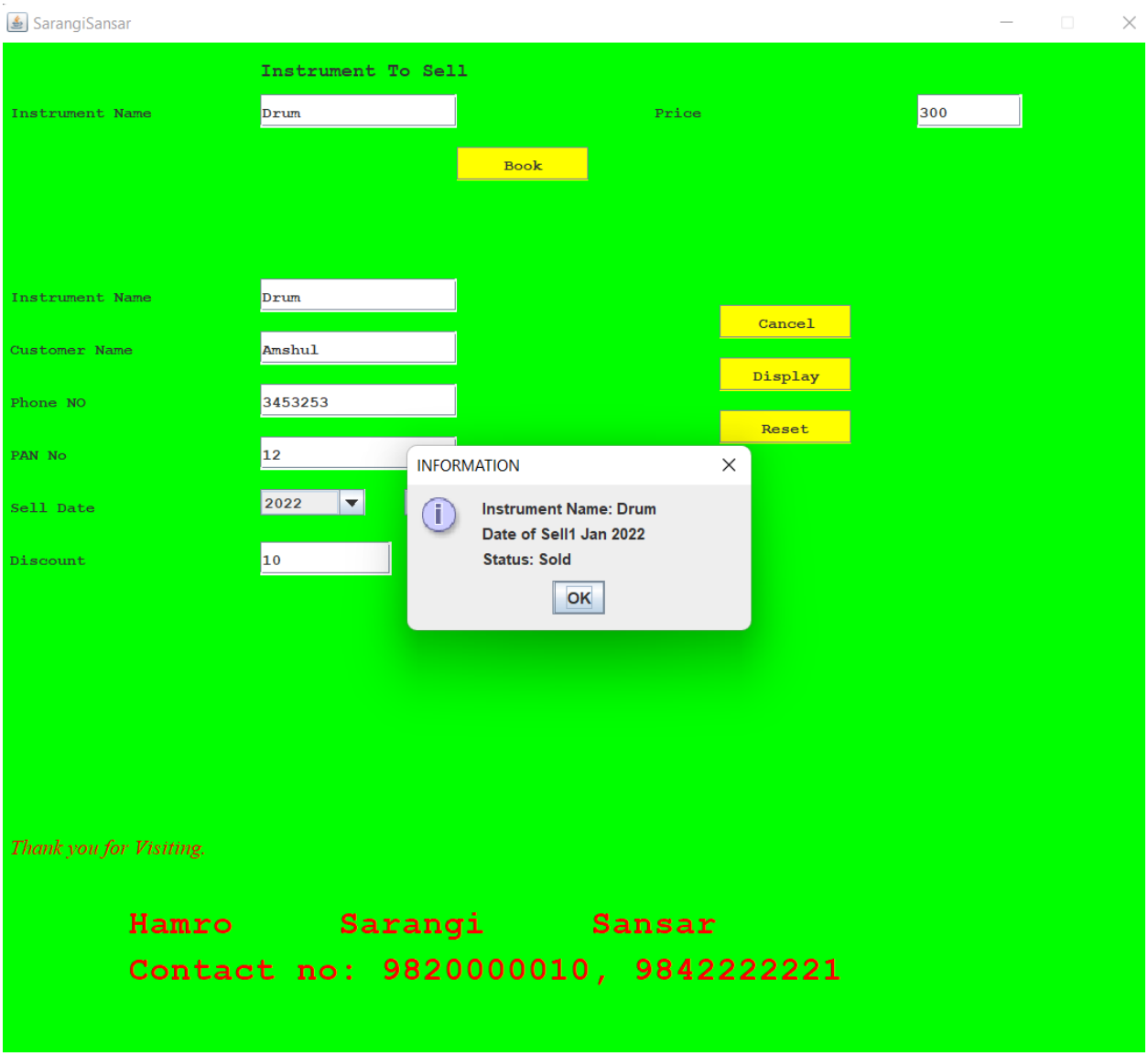


Figure 8: Sell the instrument

e) Return the instrument

Test	Return the Instrument
Objective	To return the instrument
Action	Fill the form in add panel in rent panel (third box). Click the rent button
Expected result	Message box: Successfully returned
Actual result	Message box: Successfully returned
Reference	Test successfully

Table 7: Return the instrument

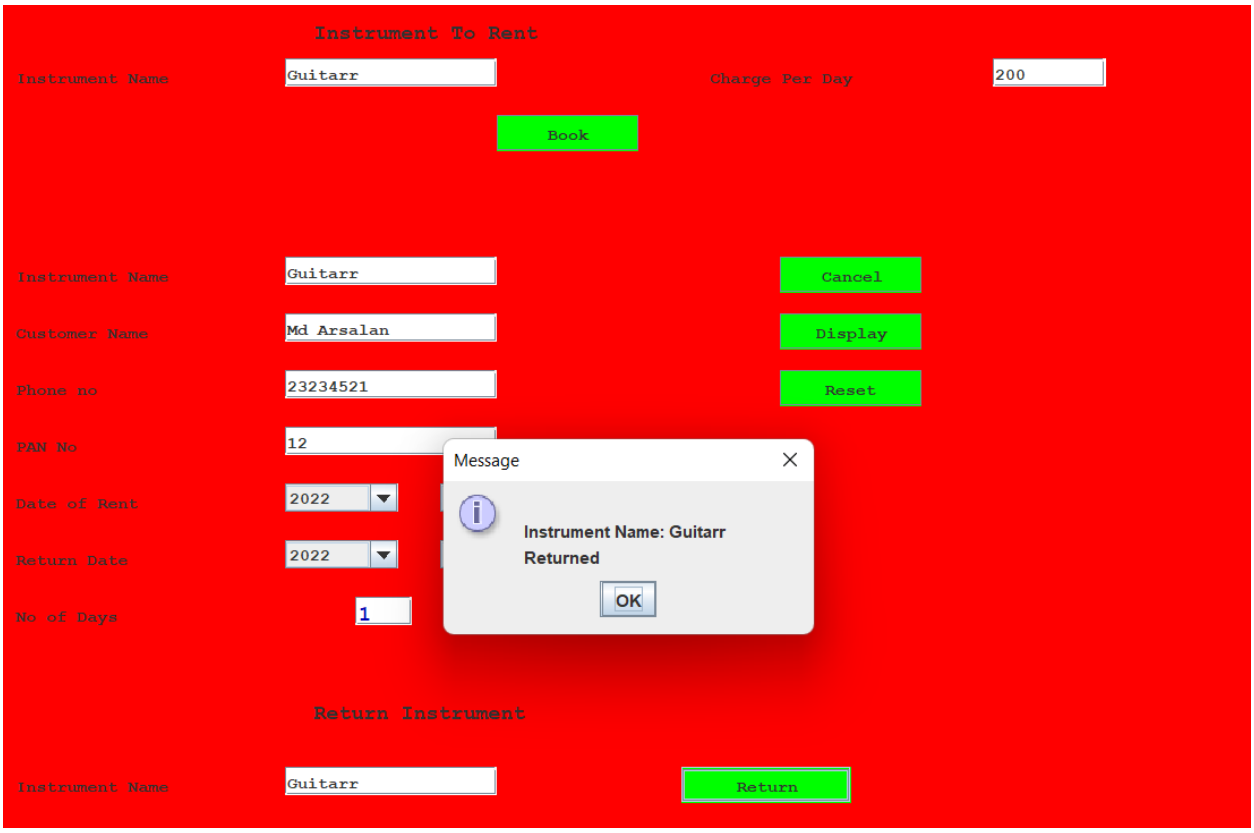


Figure 9: Return the instrument

Test 3:

I. Test 1

Action	Add the blank form or leave one text field blank
Expected Result	WARNING box: Please fill all the fields
Actual Result	WARNING box: Please fill all the fields

Table 8 Test 1: Add the blank form or leave one text field

Figure 10: Add the blank form or leave one field

II. Test 2

Action	Put string in price, pan number or discount percent instead of integer number
Expected Result	ERROR: They are invalid
Actual Result	WARNING box: Price is invalid

Table 9: Putting wrong data type

Figure 11: Putting the wrong data type

III. Test 3

Action	Add instrument for rent (first box). Leave the second box in rent panel and return the instrument
Expected Result	ERROR message: Sorry, Instrument is not rented
Actual Result	ERROR message: Sorry, Not found
Objective	Instrument cannot be returned if not rented
Reference	successful

Table 10: Condition to return the instrument

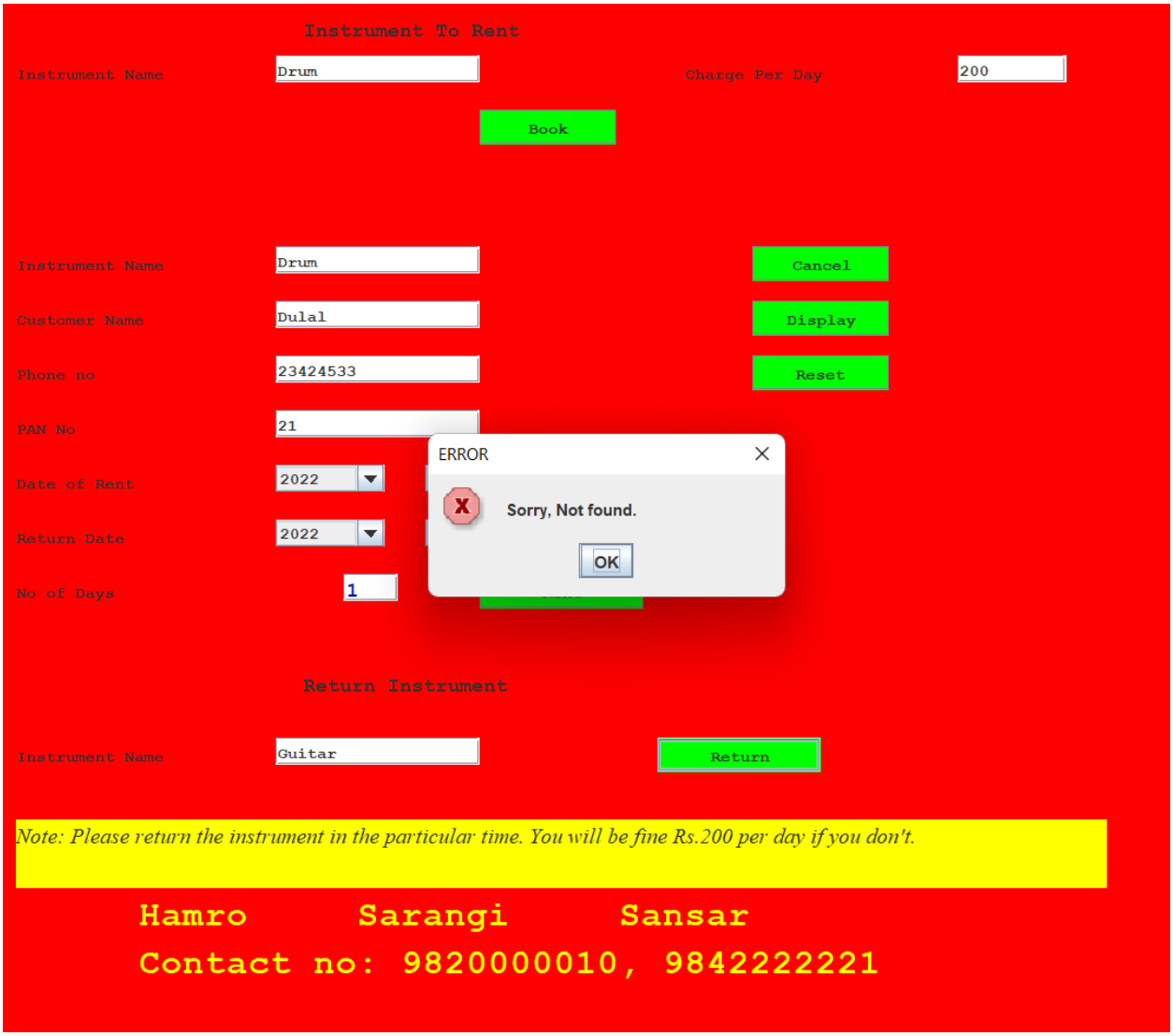
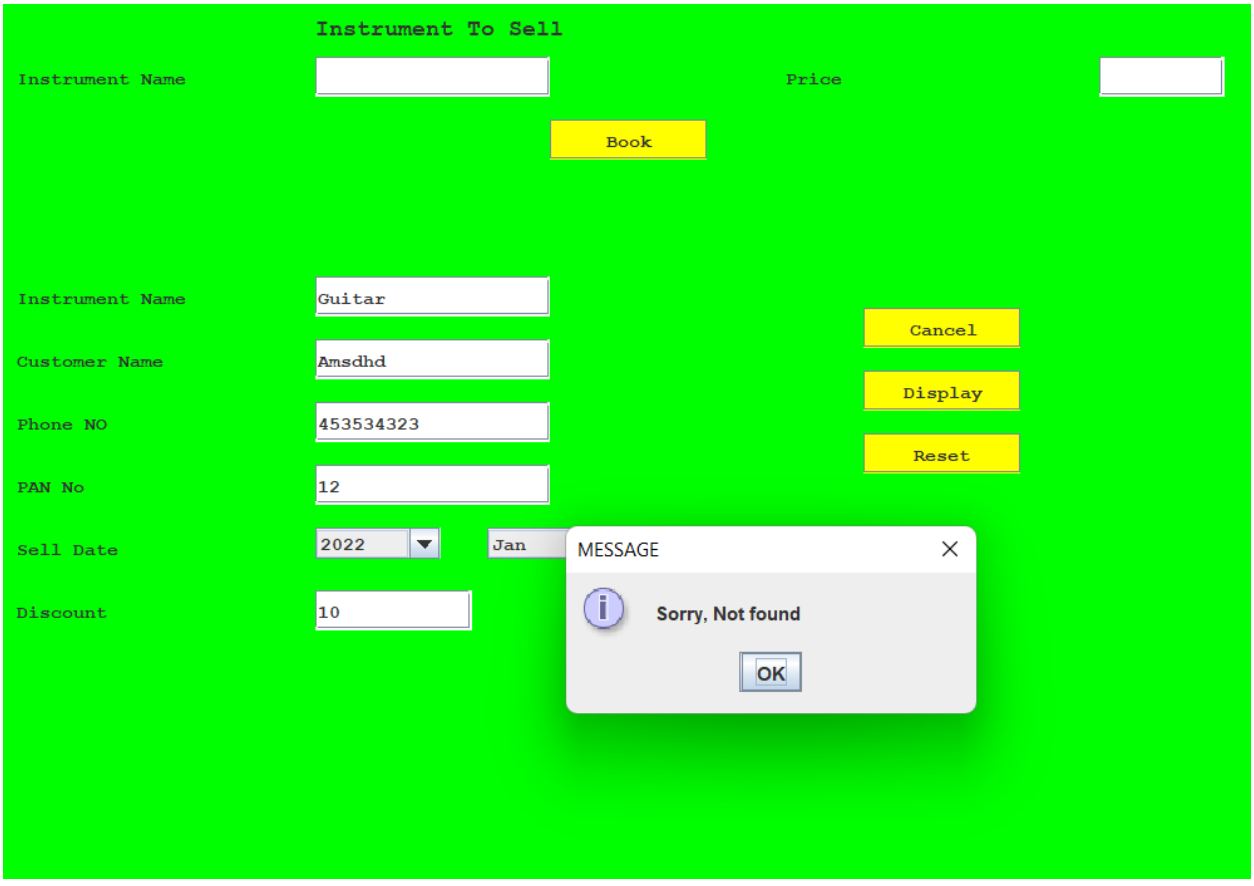


Figure 12: Condition for returning the instrument

IV. Test 4

Action	Renting the stomach without adding the instrument
Expected Result	ERROR message: Sorry, They are not found in our stock.
Actual Result	ERROR message: They are not found in our stock
Objective	Instrument cannot be rented without adding the instrument
Reference	successful



4. Error Detection and Correction

1. Syntax Error:

A syntax error in computer science refers to a mistake that a programmer makes when entering a programming language's syntax. Before the program is compiled and launched, the programmer must fix any syntax errors found by a tool called a compiler.

Programming errors known as Java syntax errors are those that happen when a programmer uses the syntax of the Java programming language; they do not include problems in the program's logic. The syntax of Java is unique, much like that of every other programming language.

For instance, one criterion for Java grammar is that every command must end with a semicolon (;).

Because I was typing quickly, I made the mistake of missing the little bracket after creating JLabel and designating it as a new variable. Same thing happened many time while creating panel, label and sometime frame. Later, when I tried to run my code, I realized it. Most programmers make this very typical error when coding in a text editor. They fail to avoid doing this by missing a minor bracket. If not, an error message will be displayed in the terminal window and the program won't run.

Error:

```
// Charge Per Day Detail in Rent
chargePday = new JLabel;
chargePday.setText("Charge Per Day");
chargePday.setFont(new Font("Arial", Font.BOLD, 20));
chargePday.setBounds(10, 50, 250, 25);
rentpanel.add(chargePday);
```

Figure 13: Syntax Error detection

Correction:

```
chargePday = new JLabel();
chargePday.setText("Charge Per Day");
chargePday.setFont(new Font("Arial", Font.BOLD, 20));
chargePday.setBounds(10, 50, 250, 25);
rentpanel.add(chargePday);
```

Figure 14: Syntax Error Correction

2. Error of datatype or Incompatible type:

This is actually not an error that I have done while programming. But also, I wanted to show why giving wrong data type can also detect error while programming or while coding. Here since I declare variable integer as a string but in the value I have written number. So, if value did not match with your data type it will throw an error in your terminal

Although it may appear to be a syntax fault, this was actually a logical issue that was found during the semantic step of compilation. The compiler's error message includes the incompatible types it has found, together with the line and location where the type mismatch occurred. (Rollbar, 2012-22)

The incompatible types issue most frequently happens when manual or explicit conversion between types is necessary, but it can also happen accidentally while utilizing a bad API, typically involving the usage of a bad reference type or the calling of a bad method with the same or similar name.

Error:

```
ArrayList<Instrument> instrumentList = new ArrayList<>();

final static int EMPTY = -1;
final static int INVALID = -2;
int num = 1;
String str = " ";
String integer = 1;
```

Correction:

```
ArrayList<Instrument> instrumentList = new ArrayList<>();

final static int EMPTY = -1;
final static int INVALID = -2;
int num = 1;
String str = " ";
int integer = 1;
```

3. Semantic Error

In general, semantics involves the usage of particular words and labels. For instance, words are used to represent network elements in a semantic network. Instead of being intended for machine interpretation, these types of semantics are targeted towards human audiences.

The semantics of computer commands may be discussed in relation to semantics in programming. Once more operating on a logical foundation, the semantic depiction of terms associated with controls, values, and other corporate branding principles. With this in mind, using terms that the machine doesn't comprehend could be referred to as a "semantic error" by programmers. When referring to instructions or chunks of code that describe objects, programmers may use the term "semantic structure."

This kind of error manifests itself during the semantic examination phase. Errors of this nature are discovered during compilation. The majority of compile-time errors are caused by scale and declaration issues. For instance, undeclared or different stated identifiers. Type mismatch is yet another compile-time error. When the incorrect adjustable is used, the incorrect operator is utilized, or the operation is carried out in the incorrect command, the semantic mistake can happen.

```
// Pan(Private Access Number) for Renting Instrument
PAN = new JLabel();
PAN.setText("PAN No");
PANN.setBounds(10, 140, 200, 25);
rentpanel2.add(PAN);
PAN.setFont(new Font("Arial", Font.BOLD, 20));
```

```
// Pan(Private Access Number) for Renting Instrument
PAN = new JLabel();
PAN.setText("PAN No");
PAN.setBounds(10, 140, 200, 25);
rentpanel2.add(PAN);
PAN.setFont(new Font("Arial", Font.BOLD, 20));
```

5. Conclusion

Finally, I would like to mention that I have successfully completed the programming coursework in the conclusion section. I had made a lot of errors, but I had also learned from them and had put my lessons into practice. I have conducted a number of searches and browsed web. I experienced numerous issues and mistakes while completing this program, which was quite difficult to discover unless I ran the application, which makes me feel a little bit uncomfortable. I'm frustrated, which causes additional mistakes and issues, yet I still manage to remain calm and got to work on my assignments. I needed more than a week to finish this coursework, but I managed to finish it on time. Since I have a background in commerce, it was a little harder for me to understand the java programming module, which had some impact on my coursework. However, I would like to thank my module teacher for making me understandable.

So, after having numerous problems with my assignment, I sought the assistance of my module teacher, who assisted me in resolving the situation. Even though the errors are small and basic, going unnoticed for a while sometimes makes me angrier and more angry, but once I realize the errors, it makes me feel ashamed. Many problems, including logical and syntactic issues, came up while I was creating the application. I made some fairly easy mistakes that prevented the program from compiling further, including forgetting to use semicolons, missing certain syntaxes like enclosing brackets, and spelling errors. However, I recognized and overcame all the faults and errors and successfully compiled the programs by inspecting and correcting them manually and, at times, with the help of the module letter. The errors really made me frustrated, which sometimes made me stop everything and just make me give up and sleep. However, it assisted me in learning from my errors and further aided in the program's development.

Due to a few issues, I only began coding eight days before the homework was due. Despite this, I did my research, and the module teacher's guidance in the tutorials and workshops greatly assisted me in finishing this project. After the program's code was finished, I began to document the curriculum, which took me longer than I had anticipated—more than three days. My physical and mental health have greatly improved as a result of this coursework, which has also enabled me to manage my frustration and, particularly in the area of coding, to spot small errors that will undoubtedly be useful to me in the future. Through this program, I learned how to handle common coding problems. In addition, I learned a lot about Java programming from this curriculum. To be completely honest, I thought this coursework would be a lot simpler, but it taught me that expectations and reality rarely match up. It was much harder than I had anticipated. It encourages me to communicate with professors more in order to get through my challenges, and it fosters a positive relationship between me and my module teacher that ultimately helps me finish the coursework. While completing this training, I overcame a lot of challenges, which taught me to never give up. So, after finishing the class, I learned a lot about java, coding, and developing a strong mentality. The primary goal of coursework is to demonstrate understanding of the material covered throughout the whole semester. After reading the syllabus, how can students use the knowledge in a practical and creative way? This is how coursework basically assesses the student's practical knowledge and creativity. Therefore, the main goal of this course is to increase students' comprehension of Java programming and show them how to use what they learn in real-world situations.

5. Bibliography

abcj, 200. *asca.* [Online] Available at: www.abc.com [Accessed 12 12 2021].

edureka, 2022. *Swing in java.* [Online] Available at: https://www.edureka.co/blog/java-swing/

PressBooks, 2022. *ISFBAB.* [Online] Available at: https://bus206.pressbooks.com/chapter/chapter-1/ [Accessed 9 1 2022].

Rollbar, 2012-22. *How to handle the incompatible data type error?.* [Online] Available at: https://rollbar.com/blog/how-to-handle-the-incompatible-types-error-in-java/#:~:text=This%20error%20implies%20that%20the,variable%20or%20method%20in%20question.

Appendix

#SarangiSansar

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.util.ArrayList;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class SarangiSansar {

    ArrayList<Instrument> instrumentList = new ArrayList<>();

    final static int EMPTY = -1;
    final static int INVALID = -2;
    int num = 1;
    String str = " ";
    int integer = 1;

    private JCheckBox adddarkmode;
```

```
    private JFrame rent, sell, homePage;

    private JPanel addInstrumenttorentpanel, panel2, rentpanel, rentpanel2,
returnPanel, addInstrumentSell, sellPanel,
            subPanel6, homepanel;
    private JLabel Rentname, instrumentRentLabel, sellTitle, chargePday,
Rentingname,
            customername, rentPhone, dateofrent, dateofreturn,
noOfDaysrented, returnofinstrumentName, PAN,
            instrumenttorentAddName, price, instrumenttosellName,
customernameSell, Sellphone, PANNo,
            dateOfSell,
            discountPercent, welcome, returnlabel;
    private JTextField rentnameTF, chargePerDayTF, rentingnameTF,
customernameTF, rentPhoneTF,
            noOfDaysrentedTF, returnofinstrumentNameTF, PANTF,
instrumentSellAddNameTF, priceTF,
            instrumenttosellNameTF,
            customernameSellTF, SellphoneTF, PANNoTF, discountPercentTF;
    private JButton addBtn, RentBtn, returnBtn, displaybuttonRent,
clearbuttonRent,
            addToSellInstrument, rentBtn, displayBtnSell, clearBtn, Back2,
frame1, frame2,
            Back;
    private JComboBox noOfDays, noOfMonths, noOfYear, noOfDaysreturn,
noOfMonthsReturn, noOfYearReturn, dayS, monthS,
            yearS;
```

```
    public SarangiSansar() {

        // date
        String days[] = { "1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "11", "12",
"13", "14", "15", "16", "17",
                "18", "19", "20", "21", "22", "23", "24", "25", "26", "27", "28", "29",
"30", "31", "32" };
        String months[] = { "Jan", "Feb", "Mar", "Apr", "May", "Jun", "July",
"Aug", "Sep", "Oct", "Nov", "Dec" };
        String years[] = { "2022", "2023", "2024", "2025", "2026" };

        // color
        Color panelColoring = (Color.WHITE);
        Color buttonColoring = (Color.WHITE);
        Color btnText = new Color(0, 0, 0);

        // font
        Font font1 = (new Font("Arial", Font.BOLD, 20));

        // Frame for Home
        homePage = new JFrame("Sarangi Sansar");
        homePage.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        homePage.setLayout(null);
        homePage.setBounds(0, 0, 690, 500);
        homePage.setLocationRelativeTo(null);
        homePage.setBackground(Color.BLACK);
        homePage.setResizable(false);
        homePage.setVisible(true);
```

```
// homePage.setAlwaysOnTop(true);

// HOME Panel
homepanel = new JPanel();
homepanel.setBounds(0, 0, 700, 500);
homepanel.setLayout(null);
homepanel.setBackground(Color.WHITE);
homePage.add(homepanel);

adddarkmode = new JCheckBox("Dark Mode");
adddarkmode.setBounds(50, 0, 150, 45);
adddarkmode.setFocusable(false);
adddarkmode.setBackground(panelColoring);
adddarkmode.setFont(new Font("Arial", Font.PLAIN, 10));
homepanel.add(adddarkmode);

adddarkmode.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent num) {

        if (integer % 2 == 0) {
            homepanel.setBackground(Color.WHITE);
            welcome.setForeground(Color.BLACK);
            adddarkmode.setBackground(panelColoring);

            integer = integer + 1;
        } else {
            homepanel.setBackground(Color.BLACK);
            adddarkmode.setBackground(Color.WHITE);
```

```
            welcome.setForeground(Color.WHITE);
            integer = integer + 1;

        }
    }

});

// Welcome Text
welcome = new JLabel("-----Sarangi Sansar-----");
welcome.setFont(new Font("Arial", Font.BOLD, 30));

welcome.setBounds(190, 50, 600, 30);

homepanel.add(welcome);

// Button for go to Sell Frame
frame1 = new JButton("Go To Sell");
frame1.setFont(new Font("Arial", Font.BOLD, 25));
frame1.setBounds(250, 200, 200, 45);
homepanel.add(frame1);
frame1.setFocusable(false);
frame1.setBackground(buttonColoring);
frame1.setForeground(btnText);
frame1.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseEntered(java.awt.event.MouseEvent num) {
        JButton hover = (JButton) num.getSource();
        hover.setBackground(Color.darkGray);
```

```
            hover.setForeground(Color.WHITE);
        }

        public void mouseExited(java.awt.event.MouseEvent num) {
            JButton hover = (JButton) num.getSource();
            hover.setBackground(Color.WHITE);
            hover.setForeground(Color.BLACK);
        }
    });

    // ActionListner for frame1 Button
    frame1.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent num) {
            homePage.dispose();
            rent.dispose();

            sell.setVisible(true);
        }

    });

    // Go to Rent Button
    frame2 = new JButton("Go To Rent");
    frame2.setFont(new Font("Arial", Font.BOLD, 25));
    frame2.setBounds(250, 280, 200, 45);
    homepanel.add(frame2);
    frame2.setFocusable(false);
```

```
frame2.setBackground(buttonColoring);
frame2.setForeground(btnText);

// Hover Effect for This Button
frame2.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseEntered(java.awt.event.MouseEvent num) {
        JButton hover = (JButton) num.getSource();
        hover.setBackground(Color.darkGray);
        hover.setForeground(Color.WHITE);
    }

    public void mouseExited(java.awt.event.MouseEvent num) {
        JButton hover = (JButton) num.getSource();
        hover.setBackground(Color.WHITE);
        hover.setForeground(Color.BLACK);
    }
});
// Actionlistner for frame2
frame2.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent num) {
        homePage.dispose();
        sell.dispose();

        rent.setVisible(true);
    }

});
```

```
// Frame for Rent
rent = new JFrame();
rent.setTitle("SarangiSansar");
rent.setLayout(null);
rent.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
rent.setBounds(20, 20, 690, 770);
rent.setResizable(false);
rent.setVisible(false);
rent.setLocationRelativeTo(null);

adddarkmode = new JCheckBox("Dark Mode");
adddarkmode.setBounds(50, 0, 150, 45);
adddarkmode.setFocusable(false);
adddarkmode.setBackground(panelColoring);
adddarkmode.setFont(new Font("Arial", Font.PLAIN, 10));
rent.add(adddarkmode);

adddarkmode.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent num) {

        if (integer % 2 == 0) {
            rent.setBackground(Color.ORANGE);
            // welcome.setForeground(Color.BLACK);
            adddarkmode.setBackground(panelColoring);

            integer = integer + 1;
        } else {
            rent.setBackground(Color.BLACK);
```

```
            adddarkmode.setBackground(Color.WHITE);
            // welcome.setForeground(Color.WHITE);
            integer = integer + 1;

        }
    }

});

// Panel for Instrument To Rent (Main Panel)
addInstrumenttorentpanel = new JPanel();
addInstrumenttorentpanel.setBounds(0, 0, 690, 770);
addInstrumenttorentpanel.setLayout(null);
addInstrumenttorentpanel.setBackground(panelColoring);
rent.add(addInstrumenttorentpanel);

// Title of This Frame
instrumentRentLabel = new JLabel();
instrumentRentLabel.setText("Instrument To Rent");
instrumentRentLabel.setFont(font1);
instrumentRentLabel.setBounds(220, 5, 300, 30);
instrumentRentLabel.setBackground(Color.black);
addInstrumenttorentpanel.add(instrumentRentLabel);

// Panel for add Instrument for Rent (Secondary Panel/Sub Panel)
rentpanel = new JPanel();
rentpanel.setBounds(10, 40, 660, 140);
rentpanel.setLayout(null);
```

```
addInstrumenttorentpanel.add(rentpanel);

// Add Instrument Name for Rent
Rentname = new JLabel();
Rentname.setText("Instrument Name");
Rentname.setFont(new Font("Arial", Font.BOLD, 20));
Rentname.setBounds(10, 10, 250, 25);
rentpanel.add(Rentname);

rentnameTF = new JTextField();
rentnameTF.setBounds(250, 10, 400, 35);
rentnameTF.setFont(font1);
rentpanel.add(rentnameTF);
// rentnameTF.setForeground(Color.BLUE.darker());

// rentnameTF.addMouseListener(new MouseAdapter(){
// @Override
// public void mouseClicked(MouseEvent num){
// rentnameTF.setText("");
// rentnameTF.setFont(font1);

// }
// });

// Charge Per Day Detail in Rent
chargePday = new JLabel();
chargePday.setText("Charge Per Day");
chargePday.setFont(new Font("Arial", Font.BOLD, 20));
```

```
chargePday.setBounds(10, 50, 250, 25);
rentpanel.add(chargePday);

chargePerDayTF = new JTextField();
chargePerDayTF.setBounds(250, 50, 400, 35);
chargePerDayTF.setFont(font1);
rentpanel.add(chargePerDayTF);

// chargePerDayTF.addMouseListener(new MouseAdapter(){
// @Override
// public void mouseClicked(MouseEvent num){
// chargePerDayTF.setText("");
// chargePerDayTF.setFont(font1);

// }
// });

// add to instrument button
addBtn = new JButton("Add");
addBtn.setBounds(250, 95, 200, 35);
addBtn.setFocusPainted(false);
addBtn.setFont(new Font("Arial", Font.BOLD, 20));
addBtn.setBackground(buttonColoring);
addBtn.setForeground(btnText);
addBtn.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseEntered(java.awt.event.MouseEvent num) {
        JButton hover = (JButton) num.getSource();
        hover.setBackground(Color.darkGray);
```

```
            hover.setForeground(Color.WHITE);
        }

        public void mouseExited(java.awt.event.MouseEvent num) {
            JButton hover = (JButton) num.getSource();
            hover.setBackground(Color.WHITE);
            hover.setForeground(Color.BLACK);
        }
    });

    // addBtn.addMouseListener(new java.awt.event.MouseAdapter() {
    // public void mouseEntered(java.awt.event.MouseEvent num) {
    // JButton hover = (JButton) num.getSource();
    // hover.setBackground(Color.RED);
    // // hover.setSize(250, 45);
    // }
    // public void mouseExited(java.awt.event.MouseEvent num) {
    // JButton hover = (JButton) num.getSource();
    // hover.setBackground(buttonColoring);
    // // hover.setSize(200, 35);

    // }
    // });

    addBtn.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent num) {

        }
```

```
});
rentpanel.add(addBtn);

// sub panel 2 /Secondary Panel for Renting Instrument

rentpanel2 = new JPanel();
rentpanel2.setBounds(10, 190, 660, 350);
rentpanel2.setLayout(null);
addInstrumenttorentpanel.add(rentpanel2);

// Now Renting Instrument
Rentingname = new JLabel();
Rentingname.setText("Instrument Name");
Rentingname.setBounds(10, 20, 250, 25);
rentpanel2.add(Rentingname);
Rentingname.setFont(new Font("Arial", Font.BOLD, 20));

rentingnameTF = new JTextField();
rentingnameTF.setBounds(250, 20, 400, 35);
rentingnameTF.setFont(font1);
rentpanel2.add(rentingnameTF);

// rentingnameTF.setForeground(Color.BLUE.darker());
// rentingnameTF.addMouseListener(new MouseAdapter(){
// @Override
// public void mouseClicked(MouseEvent num){
// rentingnameTF.setText("");
// rentingnameTF.setFont(font1);
```

```
// }
// });
// Adding Customer Detail For Rent that Instrument
customername = new JLabel();
customername.setText("Customer Name");
customername.setBounds(10, 60, 200, 25);
rentpanel2.add(customername);
customername.setFont(new Font("Arial", Font.BOLD, 20));

customernameTF = new JTextField();
customernameTF.setBounds(250, 60, 400, 35);
customernameTF.setFont(font1);

rentpanel2.add(customernameTF);

// customernameTF.setForeground(Color.BLUE.darker());
// customernameTF.addMouseListener(new MouseAdapter(){
// @Override
// public void mouseClicked(MouseEvent num){
// customernameTF.setText("");
// customernameTF.setFont(font1);

// }
// });

rentPhone = new JLabel();
rentPhone.setText("Phone no");
```

```
rentPhone.setBounds(10, 100, 200, 25);
rentpanel2.add(rentPhone);
rentPhone.setFont(new Font("Arial", Font.BOLD, 20));

rentPhoneTF = new JTextField();
rentPhoneTF.setBounds(250, 100, 400, 35);
rentPhoneTF.setFont(font1);

rentpanel2.add(rentPhoneTF);

// rentPhoneTF.setForeground(Color.BLUE.darker());
// rentPhoneTF.addMouseListener(new MouseAdapter(){
// @Override
// public void mouseClicked(MouseEvent num){
// rentPhoneTF.setText("");
// rentPhoneTF.setFont(font1);

// }
// });

// Pan(Private Access Number) for Renting Instrument
PAN = new JLabel();
PAN.setText("PAN No");
PAN.setBounds(10, 140, 200, 25);
rentpanel2.add(PAN);
PAN.setFont(new Font("Arial", Font.BOLD, 20));

PANTF = new JTextField();
```

```
PANTF.setBounds(250, 140, 400, 35);
PANTF.setFont(font1);

rentpanel2.add(PANTF);

// PANTF.setForeground(Color.BLUE.darker());
// PANTF.addMouseListener(new MouseAdapter(){
// @Override
// public void mouseClicked(MouseEvent num){
// PANTF.setText("");
// PANTF.setFont(font1);

// }
// });

// Date Of Rent YYYY/MM/DD
dateofrent = new JLabel();
dateofrent.setText("Date of Rent");
dateofrent.setBounds(10, 180, 200, 25);
rentpanel2.add(dateofrent);
dateofrent.setFont(new Font("Arial", Font.BOLD, 20));

noOfDays = new JComboBox<>(days);
noOfDays.setBounds(520, 180, 100, 30);
rentpanel2.add(noOfDays);
noOfDays.setFont(new Font("Arial", Font.BOLD, 20));

noOfMonths = new JComboBox<>(months);
```

```
noOfMonths.setBounds(360, 180, 150, 30);
rentpanel2.add(noOfMonths);
noOfMonths.setFont(new Font("Arial", Font.BOLD, 20));

noOfYear = new JComboBox<>(years);
rentpanel2.add(noOfYear);
noOfYear.setBounds(250, 180, 100, 30);
noOfYear.setFont(new Font("Arial", Font.BOLD, 20));

// Instrument Returned Date
dateofreturn = new JLabel();
dateofreturn.setText("Date of Return");
dateofreturn.setBounds(10, 220, 200, 25);
rentpanel2.add(dateofreturn);
dateofreturn.setFont(new Font("Arial", Font.BOLD, 20));

noOfDaysreturn = new JComboBox<>(days);
noOfDaysreturn.setBounds(520, 220, 100, 30);
rentpanel2.add(noOfDaysreturn);
noOfDaysreturn.setFont(new Font("Arial", Font.BOLD, 20));

noOfMonthsReturn = new JComboBox<>(months);
noOfMonthsReturn.setBounds(360, 220, 150, 30);
rentpanel2.add(noOfMonthsReturn);
noOfMonthsReturn.setFont(new Font("Arial", Font.BOLD, 20));

noOfYearReturn = new JComboBox<>(years);
noOfYearReturn.setBounds(250, 220, 100, 30);
```

```
rentpanel2.add(noOfYearReturn);
noOfYearReturn.setFont(new Font("Arial", Font.BOLD, 20));

// No of Days That Instrument is Rented For
noOfDaysrented = new JLabel();
noOfDaysrented.setText("No of Days");
noOfDaysrented.setBounds(10, 260, 200, 25);
rentpanel2.add(noOfDaysrented);
noOfDaysrented.setFont(new Font("Arial", Font.BOLD, 20));

noOfDaysrentedTF = new JTextField();
noOfDaysrentedTF.setBounds(250, 260, 400, 35);
noOfDaysrentedTF.setFont(font1);

rentpanel2.add(noOfDaysrentedTF);

noOfDaysrentedTF.setForeground(Color.BLUE.darker());
noOfDaysrentedTF.addMouseListener(new MouseAdapter() {
    @Override
    public void mouseClicked(MouseEvent num) {
        noOfDaysrentedTF.setText("");
        noOfDaysrentedTF.setFont(font1);

    }
});

// rent Button For Handle Rent
RentBtn = new JButton("Rent");
```

```
RentBtn.setBounds(250, 300, 200, 35);
RentBtn.setFocusPainted(false);
RentBtn.setFont(new Font("Arial", Font.BOLD, 20));
RentBtn.setBackground(buttonColoring);
RentBtn.setForeground(btnText);
RentBtn.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseEntered(java.awt.event.MouseEvent num) {
        JButton hover = (JButton) num.getSource();
        hover.setBackground(Color.darkGray);
        hover.setForeground(Color.WHITE);
    }

    public void mouseExited(java.awt.event.MouseEvent num) {
        JButton hover = (JButton) num.getSource();
        hover.setBackground(Color.WHITE);
        hover.setForeground(Color.BLACK);
    }
});
addBtn.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent num) {
        addinstrumenttorentBtn();
    }
});

RentBtn.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent num) {
        rentInstrument();
    }
```

```
});

rentpanel2.add(RentBtn);

// Return Panel For Return That Instrument(SubPanel)
returnPanel = new JPanel();
returnPanel.setBounds(10, 570, 660, 100);
returnPanel.setLayout(null);
addInstrumenttorentpanel.add(returnPanel);

// Data for Returning That Instrument
returnlabel = new JLabel();
returnlabel.setText("Return Instrument");
returnlabel.setBounds(250, 542, 250, 20);
addInstrumenttorentpanel.add(returnlabel);
returnlabel.setFont(new Font("Arial", Font.BOLD, 20));

returnofinstrumentName = new JLabel();
returnofinstrumentName.setText("Instrument Name");
returnofinstrumentName.setBounds(10, 18, 250, 25);
returnPanel.add(returnofinstrumentName);
returnofinstrumentName.setFont(new Font("Arial", Font.BOLD, 20));

returnofinstrumentNameTF = new JTextField();
returnofinstrumentNameTF.setBounds(250, 18, 400, 35);
returnofinstrumentNameTF.setFont(font1);

returnPanel.add(returnofinstrumentNameTF);
```

```
        // returnofinstrumentNameTF.addMouseListener(new
MouseAdapter(){
        // @Override
        // public void mouseClicked(MouseEvent num){
        // returnofinstrumentNameTF.setText("");
        // returnofinstrumentNameTF.setFont(font1);

        // }
        // });

        // return handler

        // Buttom for Return That Instrument
        returnBtn = new JButton("Return");
        returnBtn.setBounds(250, 60, 200, 35);
        returnBtn.setFocusPainted(false);
        returnBtn.setFont(new Font("Arial", Font.BOLD, 20));
        returnBtn.setForeground(btnText);
        returnBtn.setBackground(buttonColoring);
        returnBtn.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseEntered(java.awt.event.MouseEvent num) {
                JButton hover = (JButton) num.getSource();
                hover.setBackground(Color.darkGray);
                hover.setForeground(Color.WHITE);
            }

            public void mouseExited(java.awt.event.MouseEvent num) {
```

```
            JButton hover = (JButton) num.getSource();
            hover.setBackground(Color.WHITE);
            hover.setForeground(Color.BLACK);
        }
    });

    returnBtn.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent num) {
            returnInstrument();
        }
    });
    returnPanel.add(returnBtn);

    // Back Button for Back in Home Frame
    Back2 = new JButton("Go Back");
    Back2.setFont(new Font("Arial", Font.BOLD, 20));
    Back2.setForeground(btnText);
    Back2.setBackground(buttonColoring);

    Back2.setBounds(20, 682, 200, 35);
    addInstrumenttorentpanel.add(Back2);

    Back2.addMouseListener(new java.awt.event.MouseAdapter() {
        public void mouseEntered(java.awt.event.MouseEvent num) {
            JButton hover = (JButton) num.getSource();
            hover.setBackground(Color.darkGray);
            hover.setForeground(Color.WHITE);
        }
```

```
        public void mouseExited(java.awt.event.MouseEvent num) {
            JButton hover = (JButton) num.getSource();
            hover.setBackground(Color.WHITE);
            hover.setForeground(Color.BLACK);
        }
    });

    Back2.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent num) {
            rent.dispose();
            sell.dispose();
            homePage.setVisible(true);
        }
    });

    // display for Displaying Data.
    displaybuttonRent = new JButton("Display");
    displaybuttonRent.setBounds(240, 682, 200, 35);
    displaybuttonRent.setFocusPainted(false);
    displaybuttonRent.setFont(new Font("Arial", Font.BOLD, 20));
    displaybuttonRent.setBackground(buttonColoring);

    displaybuttonRent.setForeground(btnText);

    displaybuttonRent.addMouseListener(new
java.awt.event.MouseAdapter() {
        public void mouseEntered(java.awt.event.MouseEvent num) {
```

```
            JButton hover = (JButton) num.getSource();
            hover.setBackground(Color.darkGray);
            hover.setForeground(Color.WHITE);
        }

        public void mouseExited(java.awt.event.MouseEvent num) {
            JButton hover = (JButton) num.getSource();
            hover.setBackground(Color.WHITE);
            hover.setForeground(Color.BLACK);
        }
    });

    displaybuttonRent.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent num) {
            if (instrumentList.size() > 0) {
                for (int integer = 0; integer < instrumentList.size(); integer++) {
                    if (instrumentList.get(integer) instanceof InstrumentToRent)
{
                        InstrumentToRent obj = (InstrumentToRent)
instrumentList.get(integer);
                        obj.display();
                    }
                }
            } else {
                System.out
                        .println("Please fill the rent form to display your record");
            }
        }
```

```
        });
        addInstrumenttorentpanel.add(displaybuttonRent);

        // clear Button for Clearing all Data from TextField
        clearbuttonRent = new JButton("Clear");
        clearbuttonRent.setBounds(460, 682, 200, 35);
        clearbuttonRent.setFocusPainted(false);
        clearbuttonRent.setFont(new Font("Arial", Font.BOLD, 20));
        clearbuttonRent.setForeground(btnText);
        clearbuttonRent.setBackground(buttonColoring);

        clearbuttonRent.addMouseListener(new
java.awt.event.MouseAdapter() {
            public void mouseEntered(java.awt.event.MouseEvent num) {
                JButton hover = (JButton) num.getSource();
                hover.setBackground(Color.darkGray);
                hover.setForeground(Color.WHITE);
            }

            public void mouseExited(java.awt.event.MouseEvent num) {
                JButton hover = (JButton) num.getSource();
                hover.setBackground(Color.WHITE);
                hover.setForeground(Color.BLACK);
            }
        });

        clearbuttonRent.addActionListener(new ActionListener() {
```

```
    public void actionPerformed(ActionEvent num) {
        clearRent();
    }
});
addInstrumenttorentpanel.add(clearbuttonRent);

// Selling Frame for Add and sell Items
// sell
sell = new JFrame();
sell.setTitle("SarangiSansar");
sell.setLayout(null);
sell.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
sell.setBounds(0, 0, 690, 650);
sell.setResizable(false);
sell.setVisible(false);
sell.setLocationRelativeTo(null);

// panel 2/Panel For Adding Instrument To Sell
panel2 = new JPanel();
panel2.setBounds(0, 0, 690, 650);
panel2.setLayout(null);
panel2.setBackground(panelColoring);
sell.add(panel2);

sellTitle = new JLabel();
sellTitle.setText("Instrument To Sell");
sellTitle.setBounds(200, 10, 200, 20);
sellTitle.setBackground(Color.black);
```

```
panel2.add(sellTitle);

// sub panel for Adding Instrument
addInstrumentSell = new JPanel();
addInstrumentSell.setBounds(10, 40, 650, 140);
addInstrumentSell.setLayout(null);
panel2.add(addInstrumentSell);

// Instrument Name and Other Detail
instrumenttorentAddName = new JLabel();
instrumenttorentAddName.setText("Instrument Name");
instrumenttorentAddName.setBounds(20, 10, 250, 25);
addInstrumentSell.add(instrumenttorentAddName);
instrumenttorentAddName.setFont(new Font("Arial", Font.BOLD, 20));

instrumentSellAddNameTF = new JTextField();
instrumentSellAddNameTF.setBounds(250, 10, 400, 35);
instrumentSellAddNameTF.setFont(font1);
addInstrumentSell.add(instrumentSellAddNameTF);

// Price Of that Instrument
price = new JLabel();
price.setText("Price");
price.setBounds(20, 50, 200, 25);
price.setFont(new Font("Arial", Font.BOLD, 20));

addInstrumentSell.add(price);
```

```
        priceTF = new JTextField();
        priceTF.setBounds(250, 50, 400, 35);
        priceTF.setFont(font1);

        addInstrumentSell.add(priceTF);

        // add to instrument for sell button
        addToSellInstrument = new JButton("Add");
        addToSellInstrument.setBounds(250, 95, 200, 35);
        addToSellInstrument.setFocusPainted(false);
        addToSellInstrument.setBackground(buttonColoring);
        addToSellInstrument.setForeground(btnText);
        addToSellInstrument.setFont(new Font("Arial", Font.BOLD, 20));

        addToSellInstrument.addMouseListener(new
java.awt.event.MouseAdapter() {
            public void mouseEntered(java.awt.event.MouseEvent num) {
                JButton hover = (JButton) num.getSource();
                hover.setBackground(Color.darkGray);
                hover.setForeground(Color.WHITE);
            }

            public void mouseExited(java.awt.event.MouseEvent num) {
                JButton hover = (JButton) num.getSource();
                hover.setBackground(Color.WHITE);
                hover.setForeground(Color.BLACK);
            }
        });
```

```
addToSellInstrument.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent num) {
        addInstrumentForSell();
    }
});
addInstrumentSell.add(addToSellInstrument);

// sub panel 5 for
sellPanel = new JPanel();
sellPanel.setBounds(10, 210, 660, 310);
sellPanel.setLayout(null);
panel2.add(sellPanel);

// Instrument Detail For Sell Instrument
instrumenttosellName = new JLabel();
instrumenttosellName.setText("Instrument Name");
instrumenttosellName.setBounds(10, 20, 250, 25);
instrumenttosellName.setFont(new Font("Arial", Font.BOLD, 20));

sellPanel.add(instrumenttosellName);

instrumenttosellNameTF = new JTextField();
instrumenttosellNameTF.setBounds(250, 20, 400, 35);
instrumenttosellNameTF.setFont(font1);

sellPanel.add(instrumenttosellNameTF);
```

```
// Customer Detail of customer that buying Instrument
customernameSell = new JLabel();
customernameSell.setText("Customer Name");
customernameSell.setBounds(10, 60, 250, 25);
customernameSell.setFont(new Font("Arial", Font.BOLD, 20));

sellPanel.add(customernameSell);

customernameSellTF = new JTextField();
customernameSellTF.setBounds(250, 60, 400, 35);
customernameSellTF.setFont(font1);
sellPanel.add(customernameSellTF);

Sellphone = new JLabel();
Sellphone.setText("Customer Number");
Sellphone.setBounds(10, 100, 250, 25);
Sellphone.setFont(new Font("Arial", Font.BOLD, 20));

sellPanel.add(Sellphone);

SellphoneTF = new JTextField();
SellphoneTF.setBounds(250, 100, 400, 35);
SellphoneTF.setFont(font1);

sellPanel.add(SellphoneTF);

// PAN NO Detail of Sell
PANNo = new JLabel();
```

```
PANNo.setText("PAN No");
PANNo.setBounds(10, 140, 200, 25);
PANNo.setFont(new Font("Arial", Font.BOLD, 20));

sellPanel.add(PANNo);

PANNoTF = new JTextField();
PANNoTF.setFont(font1);
PANNoTF.setBounds(250, 140, 400, 35);

sellPanel.add(PANNoTF);

// Date of That Items Sold
dateOfSell = new JLabel();
dateOfSell.setText("Date of Sell");
dateOfSell.setBounds(10, 180, 200, 25);
dateOfSell.setFont(new Font("Arial", Font.BOLD, 20));

sellPanel.add(dateOfSell);

yearS = new JComboBox<>(years);
yearS.setBounds(250, 180, 100, 30);
yearS.setFont(new Font("Arial", Font.BOLD, 20));
sellPanel.add(yearS);

monthS = new JComboBox<>(months);
monthS.setBounds(360, 180, 150, 30);
monthS.setFont(new Font("Arial", Font.BOLD, 20));
```

```
sellPanel.add(monthS);

dayS = new JComboBox<>(days);
dayS.setFont(new Font("Arial", Font.BOLD, 20));
;
dayS.setBounds(520, 180, 100, 30);
sellPanel.add(dayS);

// DIscount For That Instrument
discountPercent = new JLabel();
discountPercent.setText("Discount Percent");
discountPercent.setBounds(10, 220, 200, 25);
discountPercent.setFont(new Font("Arial", Font.BOLD, 20));
sellPanel.add(discountPercent);

discountPercentTF = new JTextField();
discountPercentTF.setBounds(250, 220, 400, 35);

sellPanel.add(discountPercentTF);

// sell Button FOr Handling Data
rentBtn = new JButton("Sell");
rentBtn.setFont(new Font("Arial", Font.BOLD, 20));

rentBtn.setBounds(250, 270, 200, 35);
rentBtn.setFocusPainted(false);
rentBtn.setBackground(buttonColoring);
rentBtn.setForeground(btnText);
```

```
rentBtn.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseEntered(java.awt.event.MouseEvent num) {
        JButton hover = (JButton) num.getSource();
        hover.setBackground(Color.darkGray);
        hover.setForeground(Color.WHITE);
    }

    public void mouseExited(java.awt.event.MouseEvent num) {
        JButton hover = (JButton) num.getSource();
        hover.setBackground(Color.WHITE);
        hover.setForeground(Color.BLACK);
    }
});
rentBtn.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent num) {
        sellInstrument();
    }
});
sellPanel.add(rentBtn);

// Back Button To go Back in Home Page
Back = new JButton("Go Back");
Back.setFont(new Font("Arial", Font.BOLD, 20));
Back.setBounds(20, 550, 200, 35);
Back.setBackground(buttonColoring);
Back.setForeground(btnText);
Back.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseEntered(java.awt.event.MouseEvent num) {
```

```
            JButton hover = (JButton) num.getSource();
            hover.setBackground(Color.darkGray);
            hover.setForeground(Color.WHITE);
        }

        public void mouseExited(java.awt.event.MouseEvent num) {
            JButton hover = (JButton) num.getSource();
            hover.setBackground(Color.WHITE);
            hover.setForeground(Color.BLACK);
        }
    });
    Back.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent num) {
            rent.dispose();
            sell.dispose();

            homePage.setVisible(true);
        }

    });
    panel2.add(Back);

    // display handler for sell
    displayBtnSell = new JButton("Display");
    displayBtnSell.setBounds(240, 550, 200, 35);
    displayBtnSell.setFocusPainted(false);
    displayBtnSell.setFont(new Font("Arial", Font.BOLD, 20));
    displayBtnSell.setBackground(buttonColoring);
```

```
        displayBtnSell.setForeground(btnText);
        displayBtnSell.addMouseListener(new java.awt.event.MouseAdapter()
{
            public void mouseEntered(java.awt.event.MouseEvent num) {
                JButton hover = (JButton) num.getSource();
                hover.setBackground(Color.darkGray);
                hover.setForeground(Color.WHITE);
            }

            public void mouseExited(java.awt.event.MouseEvent num) {
                JButton hover = (JButton) num.getSource();
                hover.setBackground(Color.WHITE);
                hover.setForeground(Color.BLACK);
            }
        });
        displayBtnSell.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent num) {
                if (instrumentList.size() > 0) {
                    for (int integer = 0; integer < instrumentList.size(); integer++) {
                        if (instrumentList.get(integer) instanceof InstrumentToSell) {
                            InstrumentToSell objS = (InstrumentToSell)
instrumentList.get(integer);
                            objS.display();
                        }
                    }
                } else {
                    System.out.println("Please fill the sell form to display your sell
record");
```

```
        }
    }
});
panel2.add(displayBtnSell);

// clear handler for sell
clearBtn = new JButton("Clear");
clearBtn.setBounds(460, 550, 200, 35);
clearBtn.setFocusPainted(false);
clearBtn.setFont(new Font("Arial", Font.BOLD, 20));
clearBtn.setBackground(buttonColoring);
clearBtn.setForeground(btnText);
clearBtn.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseEntered(java.awt.event.MouseEvent num) {
        JButton hover = (JButton) num.getSource();
        hover.setBackground(Color.darkGray);
        hover.setForeground(Color.WHITE);
    }

    public void mouseExited(java.awt.event.MouseEvent num) {
        JButton hover = (JButton) num.getSource();
        hover.setBackground(Color.WHITE);
        hover.setForeground(Color.BLACK);
    }
});
clearBtn.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent num) {
        clearSell();
```

```
        }
    });

    panel2.add(clearBtn);

    subPanel6 = new JPanel();
    subPanel6.setBounds(10, 500, 140, 140);
    subPanel6.setBackground(panelColoring);
    panel2.add(subPanel6);

    sell.setVisible(false);
    rent.setVisible(false);
    homePage.setVisible(true);

}

public static void main(String[] args) {
    new SarangiSansar();
}

// add to instrument handler of Rent
public void addinstrumenttorentBtn() {
    String instrumentName = getInstrumentRentName();
    int chargePerDay = getChargePerDay();
    boolean isUnique = true;

    // negative
    if (num == 2) {
```

```
            num = 1;
            return;
        }

        // valid
        if (str == "invalid") {
            str = " ";
            return;
        }

        if (instrumentName.isEmpty() || chargePerDay == EMPTY) {
            JOptionPane.showMessageDialog(rent, "Please fill all the fields",
"WARNING",
                    JOptionPane.WARNING_MESSAGE);
            return;
        }

        for (int integer = 0; integer < instrumentList.size(); integer++) {
            if
(instrumentList.get(integer).getInstrumentName().toLowerCase().equals(ins
trumentName.toLowerCase())) {
                isUnique = false;
                JOptionPane.showMessageDialog(rent, "Instrument Name must
be unique", "ERROR",
                        JOptionPane.ERROR_MESSAGE);
                break;
            }
        }
```

```
        if (isUnique && chargePerDay != INVALID) {
            instrumentList.add(new InstrumentToRent(instrumentName,
chargePerDay));
            JOptionPane.showMessageDialog(rent, "Instrument is added to
rent successfully");
        }

    }

    // instrument name of Rent
    public String getInstrumentRentName() {
        return rentnameTF.getText().trim();
    }

    // charge per day of Rent
    public int getChargePerDay() {
        // Here, empty = -1 //invalid = -2
        String chargePerDayText = chargePerDayTF.getText().trim();
        int chargePerDay = INVALID;

        if (chargePerDayText.isEmpty()) {
            chargePerDay = EMPTY;
            return chargePerDay;
        }

        try {
            chargePerDay = Integer.parseInt(chargePerDayText);
```

```
            if (chargePerDay <= 0) {
                JOptionPane.showMessageDialog(rent, "Please enter the valid
positive number for charge", "WARNING",
                        JOptionPane.WARNING_MESSAGE);
                chargePerDay = INVALID;
                num = 2;
            }
        } catch (NumberFormatException num) {
            JOptionPane.showMessageDialog(rent,
                    "charge Per day is invalid", "Error",
                    JOptionPane.ERROR_MESSAGE);
            str = "invalid";
        }
        return chargePerDay;

    }

    // After Adding Now, Rent the instrument
    public void rentInstrument() {
        String instrumentToRentText = rentingnameTF.getText().trim();
        String customerNameRentText = customernameTF.getText().trim();
        String phoneText = rentPhoneTF.getText().trim();
        int panText = getPanNoRent();
        String dateofrent = noOfDays.getSelectedItem().toString() + " " +
                noOfMonths.getSelectedItem().toString() + " "
                + noOfYear.getSelectedItem().toString();
        String dateofreturn = noOfDaysreturn.getSelectedItem().toString() + " "
+
```

```
            noOfMonthsReturn.getSelectedItem().toString() + " "
            + noOfYearReturn.getSelectedItem().toString();
        int rentedNoOfDaysText = getrentedNoOfDays();
        boolean isUnique = true;

        if (instrumentToRentText.isEmpty() || panText == EMPTY ||
rentedNoOfDaysText == EMPTY
                || customerNameRentText.isEmpty()
                || phoneText.isEmpty()) {
            JOptionPane.showMessageDialog(rent, "Please fill all the fields",
"WARNING",
                    JOptionPane.WARNING_MESSAGE);
            return;
        }

        // negative
        if (num == 2) {
            num = 1;
            return;
        }

        // valid
        if (str == "invalid") {
            str = " ";
            return;
        }

        if (instrumentList.size() > 0) {
```

```
            for (int integer = 0; integer < instrumentList.size(); integer++) {
                if
(instrumentList.get(integer).getInstrumentName().toLowerCase()
                        .equals(instrumentToRentText.toLowerCase())) {
                    if (instrumentList.get(integer) instanceof InstrumentToRent) {
                        InstrumentToRent obj = (InstrumentToRent)
instrumentList.get(integer);
                        if (obj.getIsRented()) {
                            isUnique = false;
                            JOptionPane.showMessageDialog(rent, "Instrument is
rented already. Come Next time",
                                    "WARNING",
                                    JOptionPane.WARNING_MESSAGE);
                            return;
                        }
                        obj.rent(customerNameRentText, phoneText, panText,
dateofrent,
                                dateofreturn,
                                rentedNoOfDaysText);
                        isUnique = false;
                        obj.setIsRented(true);
                        JOptionPane.showMessageDialog(rent, "Successfully
rented");

                        break;

                    }
                }
```

```
            }
        }

        if (isUnique == true) {
            JOptionPane.showMessageDialog(rent, "They are not found in our
stock", "ERROR",
                    JOptionPane.ERROR_MESSAGE);
        }

    }

    // get pan no of That instrument in Rent
    public int getPanNoRent() {
        String panNoText = PANTF.getText().trim();
        int panNo = INVALID;

        if (panNoText.isEmpty()) {
            panNo = EMPTY;
            return panNo;
        }

        try {
            panNo = Integer.parseInt(panNoText);
            if (panNo <= 0) {
                JOptionPane.showMessageDialog(rent, "Please enter the valid
positive number pan number",
                        "WARNING",
                        JOptionPane.WARNING_MESSAGE);
```

```
            panNo = INVALID;
            num = 2;
        }
    } catch (NumberFormatException num) {
        JOptionPane.showMessageDialog(rent,
                "PAN NO. is not valid", "Error",
                JOptionPane.ERROR_MESSAGE);
        str = "invalid";

    }
    return panNo;

}

// get no of day to be Rented
public int getrentedNoOfDays() {
    String rentedNoOfDaysText = noOfDaysrentedTF.getText().trim();
    int noOfDaysrented = INVALID;

    if (rentedNoOfDaysText.isEmpty()) {
        noOfDaysrented = EMPTY;
        return noOfDaysrented;
    }

    try {
        noOfDaysrented = Integer.parseInt(rentedNoOfDaysText);
        if (noOfDaysrented <= 0) {
```

```
                JOptionPane.showMessageDialog(rent, "Please enter the valid
postive number for number of days",
                        "WARNING",
                        JOptionPane.WARNING_MESSAGE);
                noOfDaysrented = INVALID;
                num = 2;
            }
        } catch (NumberFormatException num) {
            JOptionPane.showMessageDialog(rent,
                    "Number of Days is invalid", "Error",
                    JOptionPane.ERROR_MESSAGE);
            str = "invalid";

        }
        return noOfDaysrented;

    }

    // return instrument
    public void returnInstrument() {
        String instrumentName = returnofinstrumentNameTF.getText().trim();
        boolean moreThanZero = false;

        if (instrumentName.isEmpty()) {
            JOptionPane.showMessageDialog(rent, "Please fill all the fields",
"WARNING",
                    JOptionPane.WARNING_MESSAGE);
            return;
```

```
        }
        if (instrumentList.size() > 0) {

            for (int integer = 0; integer < instrumentList.size(); integer++) {
                if
(instrumentList.get(integer).getInstrumentName().toLowerCase()
                        .equals(instrumentName.toLowerCase())) {
                    if (instrumentList.get(integer) instanceof InstrumentToRent) {
                        InstrumentToRent obj = (InstrumentToRent)
instrumentList.get(integer);
                        if (obj.getIsRented() == false) {
                            JOptionPane.showMessageDialog(rent, "Sorry,
Instrument is not rented",
                                    "ERROR",
                                    JOptionPane.ERROR_MESSAGE);
                            moreThanZero = true;
                            return;
                        }
                        obj.returnInstrument();
                        JOptionPane.showMessageDialog(rent, "Successfully
returned");
                        obj.setIsRented(false);
                        moreThanZero = true;
                        break;

                    }
                }
            }
```

```
            if (moreThanZero == false) {
                JOptionPane.showMessageDialog(rent, "Sorry, we did not rent
this or it is in stock", "ERROR",
                        JOptionPane.ERROR_MESSAGE);
            }
        }
    }

    // clear Data in Rent Frame
    public void clearRent() {
        rentnameTF.setText("");
        chargePerDayTF.setText("");
        rentingnameTF.setText("");
        customernameTF.setText("");
        rentPhoneTF.setText("");
        PANTF.setText("");
        noOfDaysrentedTF.setText("");
        returnofinstrumentNameTF.setText("");
    }

    // validation for sell
    // add to instrument handler for Sell
    public void addInstrumentForSell() {
        String instrumentName = instrumentSellAddNameTF.getText().trim();
        int price = getPrice();
        boolean isUnique = true;
```

```
        // negative
        if (num == 2) {
            num = 1;
            return;
        }

        // valid
        if (str == "invalid") {
            str = " ";
            return;
        }

        if (instrumentName.isEmpty() || price == EMPTY) {
            JOptionPane.showMessageDialog(rent, "Please fill all the fields",
"WARNING",
                    JOptionPane.WARNING_MESSAGE);
            return;
        }

        for (int integer = 0; integer < instrumentList.size(); integer++) {
            if
(instrumentList.get(integer).getInstrumentName().toLowerCase().equals(ins
trumentName.toLowerCase())) {
                isUnique = false;
                JOptionPane.showMessageDialog(rent, "Instrument Name must
be unique",
                        "ERROR",
                        JOptionPane.ERROR_MESSAGE);
```

```
                break;
            }
        }

        if (isUnique && price != INVALID) {
            instrumentList.add(new InstrumentToSell(instrumentName, price));
            JOptionPane.showMessageDialog(rent, "Instrument to sell is
added");
        }

    }

    // Instrument price for sell
    public int getPrice() {
        // empty -1 //invalid -2
        String priceText = priceTF.getText().trim();
        int price = INVALID;

        if (priceText.isEmpty()) {
            price = EMPTY;
            return price;
        }

        try {
            price = Integer.parseInt(priceText);
            if (price <= 0) {
                JOptionPane.showMessageDialog(rent, "Please enter the valid
positive number for price",
```

```
                    "WARNING",
                    JOptionPane.WARNING_MESSAGE);
            price = INVALID;
            num = 2;
          }
        } catch (NumberFormatException num) {
          JOptionPane.showMessageDialog(rent,
                "Price is  invalid", "Error",
                JOptionPane.ERROR_MESSAGE);
          str = "invalid";
        }
        return price;
    }

    // Selling That instrument
    public void sellInstrument() {
        String instrumentToSellText =
instrumenttosellNameTF.getText().trim();
        String customerNameForSellText =
customernameSellTF.getText().trim();
        String customerPhoneText = SellphoneTF.getText().trim();
        int customerPanText = getPanNoSell();
        String dateOfSell = dayS.getSelectedItem().toString() + " " +
              monthS.getSelectedItem().toString() + " "
              + yearS.getSelectedItem().toString();
        int discountPercent = getDiscountPercent();
        boolean isUnique = true;
```

```
        if (instrumentToSellText.isEmpty() || customerPanText == EMPTY ||
                discountPercent == EMPTY
                || customerNameForSellText.isEmpty()
                || customerPhoneText.isEmpty()) {
            JOptionPane.showMessageDialog(rent, "Please fill all the fields",
"WARNING",
                    JOptionPane.WARNING_MESSAGE);
            return;
        }

        // negative
        if (num == 2) {
            num = 1;
            return;
        }

        // valid
        if (str == "invalid") {
            str = " ";
            return;
        }

        if (instrumentList.size() > 0) {
            for (int integer = 0; integer < instrumentList.size(); integer++) {
                if
(instrumentList.get(integer).getInstrumentName().toLowerCase()
                        .equals(instrumentToSellText.toLowerCase())) {
                    if (instrumentList.get(integer) instanceof InstrumentToSell) {
```

```
                    InstrumentToSell objS = (InstrumentToSell)
instrumentList.get(integer);
                    if (objS.getIssold() == true) {
                        isUnique = false;
                        JOptionPane.showMessageDialog(rent, "Sorry,
Instrument is rented already", "ERROR",
                                JOptionPane.ERROR_MESSAGE);

                        return;
                    }
                    isUnique = false;
                    objS.sell(customerNameForSellText, customerPhoneText,
                            customerPanText,
                            dateOfSell,
                            discountPercent);
                    objS.setIsSold(true);
                    JOptionPane.showMessageDialog(rent, "Instrument is sold",
"INFORMATION",
                            JOptionPane.INFORMATION_MESSAGE);

                }

                break;
            }

        }
    }
    if (isUnique == true) {
```

```
            JOptionPane.showMessageDialog(rent, "They are not in or stock",
"MESSAGE",
                    JOptionPane.INFORMATION_MESSAGE);
        }
    }

    // PAN detail validation for Sell
    public int getPanNoSell() {
        String panNoSellText = PANNoTF.getText().trim();
        int panNoSell = INVALID;

        if (panNoSellText.isEmpty()) {
            panNoSell = EMPTY;
            return panNoSell;
        }

        try {
            panNoSell = Integer.parseInt(panNoSellText);
            if (panNoSell <= 0) {
                JOptionPane.showMessageDialog(rent, "Please enter the valid
positive number for pan number",
                        "WARNING",
                        JOptionPane.WARNING_MESSAGE);
                panNoSell = INVALID;
                num = 2;
            }
        } catch (NumberFormatException num) {
            JOptionPane.showMessageDialog(rent,
```

```
                "PAN NO. is invalid", "Error",
                JOptionPane.ERROR_MESSAGE);
            str = "invalid";
        }
        return panNoSell;
    }

    // discount percent FOr That Instrument
    public int getDiscountPercent() {
        String discountPercentText = discountPercentTF.getText().trim();
        int discountPercent = INVALID;

        if (discountPercentText.isEmpty()) {
            discountPercent = EMPTY;
            return discountPercent;
        }

        try {
            discountPercent = Integer.parseInt(discountPercentText);
            if (discountPercent <= 0) {
                JOptionPane.showMessageDialog(rent, "Please enter valid
positive number for discount", "WARNING",
                        JOptionPane.WARNING_MESSAGE);
                discountPercent = INVALID;
                num = 2;
            }
        } catch (NumberFormatException num) {
            JOptionPane.showMessageDialog(rent,
```

```
                "Discount Percent is invalid", "Error",
                JOptionPane.ERROR_MESSAGE);
            str = "invalid";
        }
        return discountPercent;
    }

    // clear Data in Sell Frame
    public void clearSell() {
        instrumentSellAddNameTF.setText("");
        priceTF.setText("");
        instrumenttosellNameTF.setText("");
        customernameSellTF.setText("");
        SellphoneTF.setText("");
        PANNoTF.setText("");
        discountPercentTF.setText("");
    }

}
```